AD-A221 547

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
		Final Report, 1 Nov 87 to 31 Oct 89

4. TITLE AND SUBTITLE
QUEUING NETWORKS WITH FINITE CAPACITIES

5. FUNDING NUMBERS
AFOSR-88-0028
61102F 2304/A2

6. AUTHOR(S)
Ian F. Akyildiz

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
School of Information and Computer Science
Georgia Institute of Technology
Atlanta, GA 30332

8. PERFORMING ORGANIZATION REPORT NUMBER
AFOSR-TR- 90-0464

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AFOSR/NM
Building 410
Bolling AFB, DC 20332-6448

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFOSR-88-0028

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release;
distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

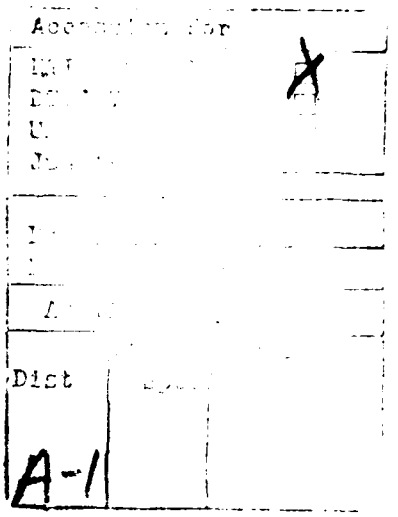
Performance has been a major issue in the design and implementation of systems such as: computer systems, production systems, communication networks and flexible manufacturing systems. The success or failure of such systems is judged by the degree to which performance objectives are met. Thus, tools and techniques for predicting performance measures are of great interest. In the last two decades it has been demonstrated several times that performance can be evaluated and/or predicted well by queuing models which can be solved either by simulation or analytical methods. Simulation is the most general and powerful technique for studying and predicting system performance. However, the high cost of running the simulation programs and uncertain statistical accuracy, makes simulation less attractive. Compared to simulation, analytical methods are more restrictive but have the advantage that it is less costly to compute numerical results. Moreover, they can be implemented very quickly, thus it is very easy to give interpretations to the relationships between model parameters and performance measures. Analytical methods have proved invaluable in modeling a variety of computer systems, computer networks, flexible manufacturing systems, etc.. They are flexible enough to represent adequately many of the features arising in such applications. They have not been able to provide much insight into the phenomenon of blocking, because all methods for networks are based on the assumption that the stations have infinite capacities. If the stations have finite capacities, blocking can occur in the network and this causes interdependencies between the stations. Hence, all the classical algorithms known in the literature cannot be applied. Therefore, in this project our aim is to solve queuing networks with finite capacities.

14. SUBJECT T

15. NUMBER OF PAGES
6

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAR



FINAL REPORT ON THE PROJECT "QUEUEING NETWORKS WITH FINITE CAPACITIES" AFOSR-88-0028

Ian F. Akyildiz

School of Information and Computer Science
Georgia Institute of Technology
Atlanta, Georgia 30332

1. Introduction

Performance has been a major issue in the design and implementation of systems such as: computer systems, production systems, communication networks and flexible manufacturing systems. The success or failure of such systems is judged by the degree to which performance objectives are met. Thus, tools and techniques for predicting performance measures are of great interest. In the last two decades it has been demonstrated several times that performance can be evaluated and/or predicted well by queueing models which can be solved either by simulation or analytical methods. Simulation is the most general and powerful technique for studying and predicting system performance. However, the high cost of running the simulation programs and uncertain statistical accuracy, makes simulation less attractive. Compared to simulation, analytical methods are more restrictive but have the advantage that it is less costly to compute numerical results. Moreover, they can be implemented very quickly, thus it is very easy to give interpretations to the relationships between model parameters and performance measures. Analytical methods have proved invaluable in modeling a variety of computer systems, computer networks, flexible manufacturing systems, etc.. They are flexible enough to represent adequately many of the features arising in such applications. They have not been able to provide much insight into the phenomenon of blocking, because all methods for networks are based on the assumption that the stations have infinite capacities. If the stations have finite capacities, blocking can occur in the network and this causes interdependencies between the stations. Hence, all the classical algorithms known in the literature cannot be applied. Therefore, in this project our aim is to solve queueing networks with finite capacities.

2. "BASIC RESEARCH" CONTRIBUTIONS

Our major contributions in this project are:

i) We were the first to develop **very efficient computational algorithms** for queueing models with finite buffers [1]. With our algorithm it is unnecessary to run long and expensive simulations for finite capacity queueing models.

ii) Extending the well-known "FLOW-EQUIVALENCY (also known Norton's Theorem) concept on finite capacity queueing models [2]. The major advantage of this technique is that computational expenses are reduced if only one or few stations from the queueing model are to be investigated under various system workloads.

iii) We also pointed out that deadlocks can occur in finite capacity queueing models with blocking [3]. We formally proved the deadlock freedom property and gave necessary and sufficient conditons for deadlock freedom.

iv) We demonstrated the application of our basic research results on several case studies [4,5,6].

Details about our contributions can be found in the attached papers [1-5].

3. APPLICATION EXAMPLES

In the following we discuss various examples from different areas such as airfleet, computer systems, communication networks, flexible manufacturing systems where our results can be applied.

3.1. Air Fleet Availability Analysis

Consider an example where we model a single flying base where operationally ready aircraft are stationed for training purposes and a repair depot where the aircraft are overhauled. The repair depot is represented as a series of stages and each stage represents the repair and replacement of a particular spare, and a repair shop and circulating spares are associated with each facility.

For example, a facility i might represent engine shop, facility ia might represent the shop for the engine and facility j, ja for radio and so on. As an aircraft leaves the flying base it always needs an engine replacement and with a probability of P_{ij} it needs a radio with P_{ija} it needs a fuel pump with P_{iia} it needs a gun sight and with probability P_{il} it needs a generator. After the repair at facility 6 it will go back to the flying base for

flight operations. Blocking can occur in this model, if for example, a spare is not available. In that case, the aircraft must wait in the according facility until the spare will be delivered.

Performance measures of the flying base, repair depot system are its productivity (total flight hours) and its operational effectiveness (availability). Availability is defined as the average fraction of aircraft available for use at a given instant. Several performance measures (e.g., average time an aircraft spends in the flying base and in the repair depot, average number of aircraft on ground at different stages of repair, use of repair facilities and average time an aircraft spends in various stages of repair) can easily be computed in the framework of our results. We can also determine the unavailability (out of stock probability) of a spare and the duration of its unavailability.

3.2. Interconnected Networks

An interconnected packet switching network consists of several local area networks which are connected by a long haul network. The local area networks are connected to a long haul network by gateways.

In the network there are two different types of communication:

i) Communication between hosts of each local area network (intranetwork traffic)

ii) Communication between hosts at different local area networks (internetwork traffic)

Hosts in the same local area network communicate with each other using a shared broadcast channel. The channel is accessed by the hosts via an interface, so-called network access unit (NAU). Based on communication protocols only one packet is allowed to be sent on the channel at a time. If a host wants to transmit a packet to another host in the same local area network it forwards it to its NAU. The access protocol of the local area network decides which packet will be transmitted next on the channel. All packets in NAU of the hosts can be seen as waiting in a global queue for accessing the channel. Once a packet obtains access to the channel it is immediately transmitted to the destination host, if source and destination hosts belong to the same local area network. If they do not belong to the same local area network, the packet is put into the NAU of the source local area network. The channel sends the packet to the gateway of the source local area network. The gateway then transmits the packet to the gateway of the destination local area network which forwards the packet to the according host through its broadcast channel.

In the queueing model the blocking occurs if a packet in any station is not allowed to leave if the destination station is full, i.e, the number of packets in the destination station is equal to its buffer capacity. In this case, the packet is blocked in the current station until a packet in the destination station is transmitted and a buffer space becomes available. Performance measures such as throughput, response time etc. can also be computed using our algorithms developed in this project. For example, we [4] applied our algorithm on this model and investigated different gateway topologies and determined their effect on the network performance.

3.3. Window Flow Controlled Packet Switching Networks

A queueing network model for a packet switching network with several virtual channels. Each virtual channel has a source and a sink. Packets in the same virtual channel follow a fixed route which may be chosen probabilistically from a finite set of routes between source and sink.

The delay for the return of an end-to-end (ETE) acknowledgement (ACK) from the sink to the source indicating receipt of a packet is modeled by an independent random variable, the distribution of which may be different for different virtual channels. This delay is modeled by an IS (infinite server) node that joins the sink to the source to yield a closed chain in the queueing network model. The flow control window size of a virtual channel is the maximum number of packets that it can have in transit within the communication network at the same time. If the number of packets in transit within a virtual channel is equal to its window size, then the source server is "blocked". A blocked source server is later unblocked when an ETE ACK returns from the sink indicating the receipt of a packet.

In [5] we demonstrated the applicability of our results on this type of model.

3.4. A Multiprocessor System Model

Consider a multiprocessor system consisting of several processors and several memory modules connected together by a multiplexed single bus. The memory modules have buffers at their inputs to queue the service requests of processors and buffers at their outputs to queue the requests served by the memory modules that cannot be served by the bus immediately. Assume a processor makes a request to a particular memory module. If the bus at that moment is not busy transferring a request for another processor or data from a memory module,

that processor takes the bus and the request is sent to that particular memory module. However, if the bus is busy transferring data, then that processor has to retry its request at a later time. If the memory module is free it will serve the request, if it is not free then the request will be queued. After the memory module completes its service, the output is placed in its output buffer for the bus, to be transmitted to the processor that made the request. The effect of a full node on its upstream nodes (nodes that have a directed arc to the full node) depends on the type of system being modeled. If the input buffers of the memory modules are full then the bus cannot place the request to the buffer, and the processor has to send a new request. The request will be transmitted a number of times until it is delivered by the bus when there is a space in the buffer. Similarly, the output buffer of a memory module can be full. In this case, the module may be forced to suspend its service until a request is delivered from its output buffer to the processor that made the request, i.e., until a space becomes available at the output buffer.

3.5. Database System Model

Users of a transaction processing system send their requests (read or update) to a request handler. When the request handler receives a request from a user, it evaluates the request and passes it to an appropriate server that is designed to handle that request type. Figure 6, being very simplistic illustrates two types of servers: one for handling inquiries and one for handling updates. The servers usually interact with the database manager to gain access to (or update) data in the database. It then formulates a reply and returns it to the user that made the requests via a reply handler.

In the queueing model of this system, the user requests are transmitted from user terminals to a request handler (REQ) over a communication channel (CC). Once the request is received and processed by the request handler, it is passed to the queue of the appropriate server: inquiry server (INQ) or update server (UP). The server processes the request and sends it to the data base manager (DBM) which accesses the disk service the request. Once the request is serviced by the disk, the data (for inquiry operation) or status (for update operation) is returned to the appropriate server. When the server completes its processing, it sends a reply to the reply handler (REP) which in turn returns the reply to the user via the communication link. The queues between various servers represent the buffers available for intermediate storage. Since there are a finite number of buffers available at each node, it is possible that one or more of the queues are full at any given time. In particular, a

user generates a request and attempts to access the communication channel. If the channel is busy transferring another request and if there is a buffer available then the request is queued. However, if there is no buffer available then the user suspends its operation, i.e., it cannot generate new requests. Similarly, the communication to the requests handler may be temporarily stopped if there is no space available in the request handler queue. Furthermore, the communication channel cannot be used to store a request due to physical constraints, hence all requests should wait in the queue until there is a space available in the requests handler's queue at which time the communication may be resumed.

REMARK. We have described several case studies where our basic research results can be applied. These case studies are representative examples. Our results can be applied to other models as well.

4. REFERENCES

1. I. F. Akyildiz, Product Form Approximations for Queueing Networks with Multiple Servers and Blocking, *IEEE Transactions on Computers*, Vol. 38, No. 1, pp. 99-115, January 1989.

2. I. F. Akyildiz and J. Liebeherr, Application of Norton's Theorem on Queueing Networks with Finite Capacities, *Proc. of the Int. Computer Networks Conf. INFOCOM 89, Ottawa, Canada, April 1989*, pp. 914-924.

3. S. Kundu and I. F. Akyildiz, Deadlock Free Buffer Allocation in Closed Queueing Networks, *Queueing Systems: Theory and Applications (QUESTA) Journal*, Vol. 4, No. 1, January 1989, pp. 47-57.

4. I. F. Akyildiz and J. Liebeherr, Gateway Performance Analysis in Interconnection Networks, *Technical Report, Georgia Institute of Technology, School of Information and Computer Science, GIT-89-027, August 1989.*

To Appear in INFOCOM'90 Computer Networks Conference, June 1990 in San Francisco.

5. I. F. Akyildiz, Performance Analysis of Computer Communication Networks with Local and Global Window Flow Control, *Proc. of the Int. Computer Networks Conf. INFOCOM 88* March 1988, pp. 400-410.